IN THE CLAIMS

Claims 1, 3, 5, 11, 14, 16, 51, 53, 55, 61, 64, 66 and 86 are amended herein. Claims 36-41 are canceled. All pending claims and their present status are produced below.

1. (Currently amended) A method for performing a transaction on a redundant database including a first database and a second database, the method comprising:

sending a set of database modifications ~~requested by~~ corresponding to the transaction to ~~a~~ the first database;

placing a message in ~~one or more~~ a message queues ~~before sending a commit command corresponding to said set of database modifications to either the first database or the second database~~, said message ~~indicating~~ comprising a representation of objects inserted, updated, or deleted in the transaction;

sending a first commit command to the first database; and;

sending said set of database modifications and a second commit command to ~~a~~ said second database.

2. (Original) The method of claim 1, further comprising:

inserting a record for the transaction into a transaction ID table in the first database.

3. (Currently amended) The method of claim 2, wherein said sending ~~a~~ said set of database modifications and said inserting are performed in the same transaction.

4. (Original) The method of claim 1, wherein the method is performed by an application server.

5. (Currently amended) The method of claim 4, further comprising:

sending a cache synchronization message to other application servers sharing ~~the~~ a same cluster as said application server.

6. (Original) The method of claim 1, wherein said set of database modifications comprises a set of structure query language (SQL) insert, update, and/or delete commands.

7. (Original) The method of claim 1, wherein said message contains a serialized representation of objects inserted, updated, or deleted in the transaction.

8. (Original) The method of claim 2, wherein said message contains a serialized representation of objects inserted, updated, or deleted in the transaction.

9. (Original) The method of claim 8, wherein said serialized representation further includes said insert of said record.

10. (Original) The method of claim 1, further comprising:

indexing messages contained in said message queue for rapid access.

11. (Currently amended) The method of claim 5, further comprising:

receiving said cache synchronization message at another application server;

extracting a transaction ID from said cache synchronization message; and

discarding message~~s~~ containing said transaction ID from ~~one or more~~ said message queue~~s~~.

12. (Original) The method of claim 2, further comprising:

periodically deleting old rows from said transaction ID table.

13. (Original) The method of claim 12, wherein said periodically deleting is performed using a background thread.

14. (Currently amended) The method of claim 5, wherein said sending said set of database modifications and ~~a~~ said commit command to ~~a~~ said second database and said sending ~~a~~ said cache synchronization message are performed asynchronously on separate threads.

15. (Original) The method of claim 5, further comprising:

detecting a failure of said first database;

halting completion of the transaction in said first database;

including in said cache synchronization message an indication that said first database is down; and

refraining from performing further actions involving said first database until said first database is restored.

16. (Currently amended) The method of claim 15, further comprising:

replaying said database inserts, updates, and/or deletes captured in said ~~cache synchronization~~ message queued in said message queue ~~at a recovery server~~ when said first database is restored.

17 (Original) The method of claim 5, further comprising:

detecting a failure of said second database;

including in said cache synchronization message an indication that said second database is down; and

refraining from performing further actions involving said second database until said second database is restored.

18. (Original) The method of claim 2, further comprising:

detecting a failure of a first recovery server;

detecting reactivation of said failed first recovery server;

reading a transaction ID out of any queued messages in a message queue corresponding to said first recovery server; and

deleting any message in said message queue that has a transaction ID matching a transaction ID in a corresponding row of said transaction ID table.

19. (Original) The method of claim 1, further comprising:

detecting a failure of a message queue;

detecting reactivation of said failed message queue;

deleting any messages in said failed message queue;
sending a message to a recovery server containing a time stamp of a first new message processed by said message queue;
receiving a message from said recovery server indicating that an oldest message still in its queue is not older than said time stamp; and
resuming normal operation upon receipt of said message from said recovery server.

20. (Previously presented) The method of claim 1, further comprising:
detecting a failure of an application server;
determining if said failure was detected during a communication with a first database or message queue;
aborting the transaction if said failure was detected during a communication with a first database or message queue;
determining if a message has been in a message queue for a predefined period of time;
discarding said message if a transaction ID for said message is not contained in a transaction ID table in said first database; and
replaying said set of database modifications to said second database if a transaction ID for said message is contained in said transaction ID table in said first database but not in a transaction ID table in said second database.

21-50. (Canceled)

51. (Currently amended) An apparatus comprising a memory and a processor for performing a transaction on a redundant database including a first database and a second database, the apparatus further comprising:
means for sending a set of database modifications ~~requested by~~ corresponding to the transaction to ~~a~~ the first database;
means for placing a message in ~~one or more~~ a message queues before sending a commit command corresponding to said set of database modifications to either the first database or the second database, said message ~~indicating~~

comprising a representation of objects inserted, updated, or deleted in the transaction;

means for sending a first commit command to the first database; and

means for sending said set of database modifications and a second commit command to ~~a~~ the second database.

52. (Original) The apparatus of claim 51, further comprising:

means for inserting a record for the transaction into a transaction ID table in the first database.

53. (Currently amended) The apparatus of claim 52, wherein said sending ~~a~~ said set of database modifications and said inserting are performed in the same transaction.

54. (Original) The apparatus of claim 51, wherein the apparatus is located on an application server.

55. (Currently amended) The apparatus of claim 54, further comprising:

means for sending a cache synchronization message to other application servers sharing ~~the~~ a same cluster as said application server.

56. (Original) The apparatus of claim 51, wherein said set of database modifications comprises a set of structure query language (SQL) insert, update, and/or delete commands.

57. (Original) The apparatus of claim 51, wherein said message contains a serialized representation of objects inserted, updated, or deleted in the transaction.

58. (Original) The apparatus of claim 52, wherein said message contains a serialized representation of objects inserted, updated, or deleted in the transaction.

59. (Original) The apparatus of claim 58, wherein said serialized representation further includes said insert of said record.

60. (Original) The apparatus of claim 51, further comprising:

means for indexing messages contained in said message queue for rapid access.

61. (Currently amended) The apparatus of claim 55, further comprising:

means for receiving said cache synchronization message at another application server;

means for extracting a transaction ID from said cache synchronization message; and

means for discarding messages containing said transaction ID from ~~one or more~~ said message queues.

62. (Original) The apparatus of claim 52, further comprising:

means for periodically deleting old rows from said transaction ID table.

63. (Original) The apparatus of claim 62, wherein said periodically deleting is performed using a background thread.

64. (Currently amended) The apparatus of claim 55, wherein said sending said set of database modifications and ~~a~~ said commit command to ~~a~~ said second database and said sending ~~a~~ said cache synchronization message are performed asynchronously on separate threads.

65. (Original) The apparatus of claim 55, further comprising:

means for detecting a failure of said first database;

means for halting completion of the transaction in said first database;

means for including in said cache synchronization message an indication that said first database is down; and

means for refraining from performing further actions involving said first database until said first database is restored.

66. (Currently amended) The apparatus of claim 65, further comprising:

means for replaying said database inserts, updates, and/or deletes captured in said ~~cache synchronization~~ message queued in said message queue ~~at a recovery server~~ when said first database is restored.

67. (Original) The apparatus of claim 55, further comprising:

means for detecting a failure of said second database;

means for including in said cache synchronization message an indication that said second database is down; and

means for refraining from performing further actions involving said second database until said second database is restored.

68. (Original) The apparatus of claim 62, further comprising:

means for detecting a failure of a first recovery server;

means for detecting reactivation of said failed first recovery server;

means for reading a transaction ID out of any queued messages in a message queue corresponding to said first recovery server; and

means for deleting any message in said message queue that has a transaction ID matching a transaction ID in a corresponding row of said transaction ID table.

69. (Original) The apparatus of claim 51, further comprising:

means for detecting a failure of a message queue;

means for detecting reactivation of said failed message queue;

means for deleting any messages in said failed message queue;

means for sending a message to a recovery server containing a time stamp of a first new message processed by said message queue;

means for receiving a message from said recovery server indicating that an oldest message still in its queue is not older than said time stamp; and

means for resuming normal operation upon receipt of said message from said recovery server.

70. (Previously presented) The apparatus of claim 51, further comprising:

means for detecting a failure of an application server;

means for determining if said failure was detected during a communication with a first database or message queue;

means for aborting the transaction if said failure was detected during a communication with a first database or message queue;

means for determining if a message has been in a message queue for a predefined period of time;

means for discarding said message if a transaction ID for said message is not contained in a transaction ID table in said first database; and

means for replaying said set of database modifications to said second database if a transaction ID for said message is contained in said transaction ID table in said first database but not in a transaction ID table in said second database.

71-85. (Canceled)

86. (Currently amended) A program storage device readable by a machine, ~~tangibly embodying~~ the program storage device containing a program of instructions executable by the machine to perform a method for performing a transaction on a database, the method comprising:

sending a set of database modifications ~~requested by~~ corresponding to the transaction to ~~a~~ the first database;

placing a message in ~~one or more~~ a message queues before sending a commit command corresponding to said set of database modifications to either the first database or the second database, said message ~~indicating~~ comprising a representation of objects inserted, updated, or deleted in the transaction;

sending a first commit command to the first database; and;

sending said set of database modifications and a second commit command to ~~a~~ the second database.

87-91. (Canceled)